CALL ADDMISSION CONTROL SYSTEM AND METHOD FOR INTERPRETING SIGNALING MESSAGES AND CONTROLLING TRAFFIC LOAD IN INTERNET PROTOCOL DIFFERENTIATED SERVICES NETWORKS

5

BACKGROUND OF THE INVENTION

Technical Field of the Invention

10   The present invention relates to a call admission control (CAC) system and method for Internet Protocol (IP) Differentiated Services (DiffServ) network having at least one node for interpreting signaling messages and controlling traffic load in the network. In particular, and not by way of limitation, the present invention is directed to a system

15   and method for interpreting signaling messages and controlling traffic load in IP network of Universal Mobile Telecommunication System Terrestrial Radio Access Networks (UTRAN) using only functionalities implemented in the IP network layer and the underlying layers.

20

Description of Related Art

IP networks were designed originally for best effort (BE) data services. Recently, there has been increasing demand to use IP networks for transmitting real-time traffic

25   like voice, multimedia, or other delay-sensitive and jitter-sensitive traffic types. There are also plans to use IP in UTRAN as a transport protocol in mobile access networks, where, due to the mobility of the users, there are strict delay requirements and other Quality-of-Service (QoS)

30   requirements for all traffic types. In IP UTRAN, real-time applications generate a large portion of the traffic. Thus, providing QoS guarantees (delay and loss) to real-time traffic is one of the most important requirements. As

overload – either call level or packet level – in the system
results in too large delay for most of the packets, it is
essential to include functions that prevent overload
situations. To protect against call level overload,
5   therefore, the CAC algorithm is crucial.

DiffServ in IP-UTRAN

The current Third Generation Partnership Program (3GPP)
agreements define the requirements against the IP UTRAN
10 Transport Network Layer (TNL), but do not specify the way
the TNL actually implements QoS. The QoS differentiation
provided by the TNL can be based either on hop-by-hop or on
end-to-end basis, and the TNL may support either flow-per-
flow or aggregate classification. The necessary information
15 for QoS differentiation among UTRAN flows is provided by the
Radio Network Layer (RNL).

The transport network should be able to handle both
UTRAN traffic and non-UTRAN traffic. Thus, existing QoS IP
solutions should be used in the QoS architecture of the
20 transport network. Due to scalability reasons, the DiffServ
(DS) concept is preferred, nevertheless the IP UTRAN concept
and the DiffServ concept differ in some important points.

The DiffServ architecture was originally designed for
Internet backbones, which implies that several network
25 providers own and operate it. The DS network, therefore, is
divided into domains. At the boundaries of DS domains,
traffic is regulated to allow e.g. charging and the
operation of the internal of the domain. The parameters of
services that a domain offers are described in the Service
30 Level Specification (SLS), which includes a Traffic
Conditioning Specification (TCS), which specifies how
traffic conditioners should be configured.

The most frequently mentioned design goal of DiffServ
is scalability. The large functional difference between
(complex) boundary and (simple) interior nodes is due to
this criterion. Traffic conditioning is one of the roles of
5   boundary nodes, which is to measure the incoming flows and
to ensure that they conform to the SLS/TCS. Boundary nodes
are also responsible for marking/re-marking of incoming
packets according to the Per Hop Behavior (PHB) that the
flow requires inside the DiffServ domain. In contrast to
10 this, interior nodes are typically not required to do
traffic conditioning. Their task is to forward packets
according to the required PHB signaled in the DS field of
the IP header. Note that a PHB describes a special type of
requirements demanded by flows, but it does not say anything
15 about the exact mechanisms (such as scheduling method,
buffer management, or policing method) that the router
should include. It is up to the router designer how the
router supports a specific PHB.

According to the DiffServ concept, neither boundary
20 nodes nor interior nodes support CAC. If the traffic at a
given ingress node i.e. the boundary node where the traffic
enters the DiffServ domain exceeds the volume set in the
SLS/TCS, then packets are dropped due to policing. However,
policing is applied to the aggregate traffic, thus it
25 degrades the performance of many real-time applications, as
opposed to CAC, where the integrity of admitted connections
is always preserved. Having recognized this fact, several
solutions evolved proposing a session level control plane to
the aggregate user plane of DiffServ and also proposing
30 integration of Integrated Services (IntServ) and DiffServ
networks.

RFC 2998 proposes a framework for supporting IntServ
over DiffServ networks. According to the RFC, the IntServ

capable part of the network includes RSVP-aware nodes, which inherently include per-session states in the forwarding and signaling planes. Edge nodes, which are at the edge of the IntServ capable part of the network, do call admission
5   control on behalf of the DiffServ region, which does not maintain per-session states. Thus, DiffServ regions of the network are treated as virtual links connecting IntServ capable routers or hosts from the perspective of IntServ.

A next step in the evolution of DiffServ networks is
10   when interior nodes also support a resource reservation signaling protocol, such as Resource Management in DiffServ (RMD). These solutions keep the user plane of DiffServ routers, which does not have per-flow separation, and add resource reservation to that.

15

Static Provisioning in DiffServ

In the current DiffServ architecture, no resource reservation signaling protocol is implemented within a DS domain. That is, interior nodes do not register ongoing
20   connections and do not implement any admission control functionality. Furthermore, boundary nodes do not get any feedback about congestion in interior nodes.

To avoid overload, the bandwidth allocation inside the DS domain is static. Bandwidth allocation is typically
25   implemented by two functions in DS routers, such as scheduling and policing. Scheduling, specifically weighted fair queuing (FQ), is a means for allocating a minimum guaranteed bandwidth for a given DS class. Policing, on the other hand, guarantees that the maximal bandwidth used by a
30   given class is also limited.

Regarding dimensioning and admission control, a trunk reservation model can be used. A trunk is a virtual capacity allocated for flows with the same ingress (boundary node

where traffic enters the DS domain) and egress (boundary
node where traffic leaves the DS domain) nodes. Ingress
nodes have to ensure – via admission control – that the
aggregate (effective) bandwidth of flows in a trunk does not
5   exceed the assigned trunk capacity. In other words, if the
aggregate bandwidth of flows with the same ingress-egress
boundary node – including the new request – would exceed
the capacity of the corresponding trunk then the new request
must be blocked. Once the ingress node admits a connection,
10   other nodes in the domain cannot block it.

The following table summarizes the functions used in a
statically provisioned network.

	Function	Static provisioning
INTERIOR NODES	Bandwidth allocation to DS classes	Static
	Configuration of traffic conditioners	Static
	Resource reservation protocol	No
	Admission control	No
BOUNDARY NODES	Admission control	Yes
	Resource reservation protocol	Yes
	Configuration of traffic conditioners	Static

The trunk reservation idea can be extended to IP UTRAN
15   systems where flows have strict delay requirements and there

are multiple traffic classes. Two solutions can be applied in this context, which are as follows.

Multiple single-class trunks

To construct a trunk reservation model in DiffServ, an apparent solution is to completely separate the handling of DiffServ classes in the admission control. That is, between an ingress-egress pair each QoS class has separated trunks. This solution is inline with the DiffServ concept, as it requires static bandwidth allocation for QoS classes in all nodes, including interior and boundary nodes (routers).

Single multi-class trunk for each ingress-egress pair

The previous static provisioning method completely separates the resource reservation of flows having the same ingress-egress node but different DS class. By allowing statistical multiplexing between all flows within an ingress-egress pair, significant capacity gain can be achieved. This can be done by allocating a multi-class trunk between each ingress-egress pair of boundary nodes, as shows, instead of several single-class trunks as in the previous method.

Statistical QoS requirements of UTRAN traffic may be violated due to overload and due to short time-scale bursts in a normal (non-overloaded) situation. Classical CAC methods consider only overload caused by the on-off behavior of sources, but they do not protect against unacceptably long waiting times due to short time-scale bursts.

Resource allocation problem in UTRAN with ATM as transport network was investigated in Sz. Malomsoky, S. Rácz and Sz. Nádas, "Connection Admission Control in UMTS Radio

Access Networks," Computer Communications - Computer Communications 26 (2003) p. 2011-2023. This model was similar to the trunk model of IP UTRAN. However, they did not consider any QoS differentiation, that is, traffic was 5 served in a common FIFO-queue. Based on analytic results, they presented a simple Call Admission Control method, which considers statistical delay requirements of UTRAN traffic.

This CAC method can be directly applied for IP UTRAN in case of single-class trunks, if large IP packets are 10 segmented to sufficiently small segments. The main idea of this solution is to handle each DiffServ class separately. Separation means that the actual load of other QoS classes is not used in the call admission control. That is, regarding the load of other classes a worst case assumption 15 has to be applied, i.e. they are assumed to be overloaded. When other classes are overloaded, the studied class can be handled as if it was a stand-alone FIFO system of which capacity is equal to the allocated bandwidth.

The waiting times in a DS queue in case of multi-class 20 trunks are hard to determine analytically. However, a basic CAC approach can be presented that is based on the worst-case value of the service rate of each queue, which allows us to model each real-time queue as a separated FIFO-system in the network node. Under this assumption (referred as 25 Separated FIFO model), CAC method for FIFO systems can be applied for multi-class trunks.

Thus, although the above-mentioned methods are suitable for reducing bandwidth capacity needs, each of them has 30 disadvantages that limit their applicability. It would be desirable to have a method for IP networks that achieves a better utilization of the admissible region of the IP network in an efficient manner. Such a method would use only

functionalities implemented in the IP network layer and the underlying layers, and would utilize an algorithm that is simple and fast. The method would not require high processing capacity, and it would be easy to implement. The

5   present invention provides such a method.

SUMMARY OF THE INVENTION

In one aspect, the present invention is directed to a method for IP DiffServ networks having at least one node for

10   interpreting signaling messages and controlling traffic load in the network. The method consists of an initialization and a real-time phase. In initialization phase, coefficients of the approximating hyperplanes are computed and stored. This phase is repeated when the descriptor of a traffic class

15   changes, which usually happens when a node is configured or reconfigured. In the real-time phase a traffic mix is admissible, if for each real-time traffic class both the stability and the delay constraints are fulfilled.

In another aspect, the present invention is directed to

20   a system for IP DiffServ networks having at least one node for interpreting signaling messages and controlling traffic load in the network. The system includes functions for computing coefficients of approximation hyperplanes, storing the coefficients of approximating hyperplanes, determining

25   whether descriptor of a traffic class changed, determining whether an admission request is accepted, estimating the loss ratio and comparing the minimal allowed tolerance level in the queue; comparing the scalar product of the traffic mix vector and the normal vector of the hyperplane to the

30   array of the effective bandwidth value, admitting traffic mix if for each real-time traffic class both the stability and the delay constraints are fulfilled.

Yet another aspect of the invention is a node having means implementing the functions mentioned above.

The CAC method described above makes possible to
5   utilize statistical multiplexing gain in multi-class trunks in a static provisioned IP-UTRAN network. This leads to the advantage that the effective bandwidth need of a traffic mix transferred in a multi-class link will be lower than the sum of effective bandwidth needs of single-class links used for
10   transferring the same traffic mix (packet level gain). Another advantage of the present invention is that a DS class could use more bandwidth than allocated when other classes are under-utilized (call level gain).

15   **BRIEF DESCRIPTION OF THE DRAWINGS**

For a more complete understanding of the prior art and the invention, reference is made to the following detailed description taken in conjunction with the accompanying drawings wherein:
20   Figure 1 is an exemplary simplified block diagram of the architecture of a per DS class trunk reservation in IP UTRAN according to the prior art;

Figure 2 is an exemplary simplified block diagram of the architecture of a single multi-class trunk per ingress-
25   egress pair in IP UTRAN according to the prior art;

Figure 3 - 4 are graphs showing admissible region of an ingress-egress trunk with DS class separation and without separation;

Figure 5 is a graph showing simulative results for delay
30   constraint surface;

Figure 6 is a schematic diagram of the Separated Strict Priority model derivation;

Figure 7 is a graph of hyperplanes of class -1 delay constraint surface for a three-class case;

Figure 8 is flow chart depicting the two phases of the admission control process;

5   Figure 9 is an exemplary flowchart of the real-time phase of the admission control process.

DETAILED DESCRIPTION OF THE INVENTION

Figure 1 is an exemplary simplified block diagram of the
10   architecture of IP UTRAN that includes a system with four base stations 1, five boundary routers 2, two interior routers 3, an RNC 4 and four multiple single-class trunks 5. There are three DS classes forming the trunks 5 indicated by solid lines, and thus trunks 5 originate from each boundary
15   router 2 that is connected to a base station 1.

Figure 2 is an exemplary simplified block diagram of IP UTRAN showing the same architecture of base stations 1, boundary routers 2, interior routers 3 and RNC 4. Unlike
20   Figure 1, four single multi-class trunks 5 are applied for each ingress-egress pair. The capacity allocations for QoS classes are static within a multi-class trunk 5. This can be ensured by any class based scheduling method in the boundary routers 2, such as class based weighted fair queuing.
25   Opposite to the previous method, unused bandwidth of a DS class can be used by other classes. The capacity gain comes from two reasons. Firstly, the effective bandwidth need of a traffic mix transferred in a multi-class trunk will be lower than the sum of effective bandwidth needs of single-class
30   trunk used for transferring the same traffic mix (packet level gain). Secondly, a DS class could use more bandwidth than allocated when other classes are under-utilized (call level gain).

Figure 3 and 4 show the admissible region of an ingress-egress trunk with and without DS class separation. In case of multi-class trunks, bandwidth not used by a queue is
5   redistributed among the others. Thus, service rate of a queue may be substantially higher than its guaranteed minimum rate, hence the Separated FIFO model may result too pessimistic approximation for the queuing delay. This is the typical case if multiple real-time classes are present in
10   the system.

The admissible region of a multi-class trunk with two QoS classes and another one with three QoS classes is shown in the Figures where bold lines indicate the maximal available admissible region while dashed lines show the
15   admissible region in case of FIFO-approximation. It can be seen that there is a large difference in the admissible regions.

Figure 5 is a graph showing simulative results for
20   delay constraint surface. In order to give a better approximation of the service-rate of a real-time queue than the Separated FIFO model, bandwidth redistribution among DS queues should be considered. Therefore, a set of scheduling models is constructed where the approximation of the service
25   rate is better, but still conservative, thus the queuing delay in the models are an upper bound for the queuing delay in the original FQ system.

The delay constraint surface of a given class i is defined as the maximal number of class i sessions with
30   respect to their delay requirements in function of number of sessions in other classes. The scheduling models provide a conservative approximation for the delay constraint surfaces. The Figure shows the delay constraint surface of a

given class in a three-class FQ system, which is obtained by simulations. Each class has its own queue in the scheduler in this case. It can be observed that the surface is non-linear and decreasing. It reaches its maximum when all
5   ·classes but the observed one has no sessions. Furthermore, it never falls below a certain level even if other classes generate high load. The absolute minimum level corresponds to the number of admissible connections given by the Separated FIFO model. In other words, the Separated FIFO
10   model approximates the delay constraint surface with a horizontal plane that is fitted to its minimum value.

     Before introducing the models, the service-rates in the original FQ system is analyzed if a traffic mix is given.
15   Depending on the traffic mix, some of the queues may be saturated, that is, they are continuously utilizing their service rates. Other queues, i.e. the non-saturated queues, are empty in a part of the time and utilize their services rates only in the busy periods. Notations S and S^c are used
20   for the set of non-saturated and saturated queues, respectively. The service rate of a non-saturated queue i is observed. Firstly, the case should be considered where each non-saturated queue but i is empty. In this case, link rate is shared among saturated queues and queue i, according to
25   their weights. Thus, service rate of queue i is $\dfrac{w_i}{w_i + \sum_{j \in S^c} w_j} C$.

     When a busy period in a non-saturated queue k begins, e.g. a packet arrives in queue k, then the system ensures service rate to queue k by reducing service rates of other queues, according to their weights. Thus, $\dfrac{w_i}{w_i + \sum_{j \in S^c} w_j}$ share of the
30   service rate of queue k comes from queue i. If busy periods in additional non-saturated queues begin, then the system

also ensures their service rates by further reducing the service rates of other queues. Therefore, service rate of queue i varies in time between $\dfrac{w_i}{w_i + \sum_{j \in S^c} w_j} C$ and $\dfrac{w_i}{w_i + \sum_j w_j} C$ depending on the arrival process in non-saturated queues.

5   To construct a scheduling model which provides a conservative approximation of service rate process of queue i, the following simplifications can be taken. Firstly, saturated queues are separated from the scheduler. The linkrate of the reduced system is $\dfrac{w_i}{w_i + \sum_{j \in S^c} w_j} C$, which is an

10 upper bound to the service rate of queue i. Secondly, it can be assumed that $\dfrac{w_i}{w_i + \sum_{j \in S^c} w_j}$ share of bandwidth needs of non-saturated queues are ensured from the reduced system. Accordingly, the arrival processes of non-saturated queues are directed into the reduced system, but size of the

15 packets is reduced to $\dfrac{w_i}{w_i + \sum_{j \in S^c} w_j}$ proportion of the original size. Thirdly, the packet service order is realized in the original system depends on the weight settings and on the actual FQ implementation. To give a worst-case approximation, it can be assumed that packets in the non-

20 saturated queues are served as if they had higher priority than packets in queue i. As a result, the reduced system works as a Strict Priority scheduler, therefore this model can be called as a Separated Strict Priority model.

25   Figure 6 shows the model in case of a three queue 61, 62, 63 system where the observed queue is queue 62 and queue 61 is saturated ($S^c = \{1\}$). The Separated SP model 65 gives a good approximation of the service rate of queue i if the utilization of the non-saturated queues (except queue i) is

low. Nevertheless, it is conservative approximation at any
traffic mixes. Therefore, queuing delay of queue i packets
in the model is an upper bound for the queuing delay in the
original system. Depending on which real-time (RT) queues

5   are saturated, 2^{L-1} different models can be devised, where L
is the number of RT queues. Different models give the best
approximation   for   different   traffic   conditions.   The
Separated FIFO model is a special case of this model where
each queue but i is saturated. Note, that traffic in the

10   best effort (BE) queues are not controlled, hence BE queues
are considered to be always saturated.

By using the Separated SP models 65, the approximation
problem of delay constraint surface in the FQ system 64 was

15   simplified   to   the   problem   of   approximating   the   delay
constraint surface of low priority (LP) classes in multi-
class   strict   priority   systems.   Although   the   detailed
analysis of strict priority systems is beyond the scope of
this   invention,   a   simple   method   can   be   given   for   the

20   approximation. Via simulations can be shown that in a two-
class SP scheduler with periodic input the delay constraint
curve of the LP class is close to linear and it is convex if
packets   are   segmented.   The   linear   approximation   can   be
extended to multi-class strict priority systems. Although no

25   analytic   result   is   known   which   describes   the   whole
constraint   surface,   but   our   simulation   results   on   a   wide
range of parameters at two- and three-class cases indicate
that   the   delay   constraint   surface   of   a   LP   class   can   be
approximated by a single hyperplane, and this approximation

30   is conservative.
Denote S the index set of queues and K^S the number of
classes in the SP system. An approximation hyperplane is
given in the space of number of sessions, if K^S points of the

hyperplane are known. Therefore, the points of the approximation hyperplane of LP class i can be determined where only a single class i session is in the system and all other classes but j is empty (no ongoing sessions are

5   present from other classes). Accordingly, P_{ji}^{S} is defined as the maximal number of class j sessions in the SP system if delay requirement of a single class i session should be kept and all other classes are empty.

The complementary distribution function of queuing delay

10 seen by a class i packet arriving at a random time instance in the SP system fed by number of n_j class-j sources is denoted by $D_C^{n_j}(t)$. Considering the on-off behaviour of class-j traffic, the maximal number of class-j sources with respect to the delay requirements in class-i is

15   (1) $P_{ji}^{S} = \max\left\{ N_j \mid \sum_{n_j=0}^{N_j} D_C^{n_j}\left(d_i - \frac{b_i}{C}\right) \cdot \Pi(n_j) < \varepsilon \right\}$

where $\Pi(n)$ is the probability that n from the N class j sessions are active, C is the link capacity and ε is the tolerance level of QoS violation. We can distinguish two cases depending on whether class i and j is served in the

20 same queue or not.

Firstly, if class i and class j is served in the same queue, this corresponds to a two class FIFO system. Then the queuing delay seen by a class i packet can be expressed by the steady-state virtual waiting time distribution of the

25 FIFO queue. Formulas for virtual waiting time distribution in a FIFO-queue fed by periodic sources are well-known. Thus, P_{ji}^{S} can be determined iteratively from (1) or directly by using approximations.

Secondly, the case is considered when class i and j

30 served in different queues, that is, class j packets have priority over class i packets. The equation of an

approximating hyperplane of the delay constraint of class i then can be written as

$$(2) \quad \sum_{j \in S} E_{ji}^S \cdot N_j = P_{ii}^S + 1$$

where $E_{ji}^S = P_{ii}^S / P_{ji}^S$. In the special case, when no session
5   from class j can be admitted without violating QoS requirements of class i, that is, $P_{ji}^S = 0$, we set $E_{ji}^S = P_{ii}^S + 2$ to avoid division by zero. This guarantees that the corresponding hyperplane remains conservative.

10   As it was seen in the previous subsections that 2^{L-1} different Separated SP models can be constructed for a FQ system with L real-time queues, which conservatively approximate the service rate of a class. Accordingly, 2^{L-1} different hyperplanes can be determined, and all of them
15   conservatively approximate the delay constraint surface of that class in the original system. We use the linear approximation of the delay constraint surface to check whether delay requirements of a given class will be kept if the traffic mix is admitted. If the traffic mix is below at
20   least one of the approximating hyperplanes in the space of number of sessions, then the specific delay constraint is fulfilled. This is tested by checking

$$(3) \quad \max_S \left(P_{ii}^S + 1 - \sum_{j \in S} E_{ji}^S \cdot N_j \right) > 0$$

This test should be run for each real-time class.
25   Therefore, the computational complexity of the real-time phase is $O\left(K^2 \cdot 2^{L-1}\right)$ if the number of real-time classes is K and that of real-time queues is L. However, the complexity of initialization and real-time phase can be decreased if not all of the approximating hyperplanes are determined. The
30   relevant hyperplanes, which are the ones that admit the most

connections at frequent traffic mixes, should only be
determined. Queues that usually contain many connections can
be taken as saturated and queues with few connections as
non-saturated, which gives only a single approximating
5   hyperplane for delay constraint regions. If each delay
constraint region is approximated with a single hyperplane,
then the complexity of the real-time phase is $o(K^2)$, which is
the same as with the Separated FIFO model. Note that at
least one hyperplane is necessary for each delay constraint
10   to ensure correct operation.

Figure 7 demonstrates the linear approximation of the
delay constraint surface of class 1 in a three-class system
where each class has its own queue. Plane 1 belongs to the
15   case when none of the queues is saturated. Plane 2 and Plane
3 correspond to the case when one of the queues is saturated
and finally at Plane 4 both queue 2 and queue 3 are
saturated. The approximation of the delay constraint region
of class 1 can be obtained as the union of the regions
20   bordered by the approximating surfaces.

Besides short time-scale bursts, delay violation caused
by overload should be also avoided. A queue is considered
overloaded, if the mean input rate in that queue exceeds its
25   mean service rate. The ratio of lost packets, which is a
part of the QoS measure, expresses the probability that a
packet arrives in an overloaded situation. Direct
calculation of the loss ratio is a computationally complex
task if the number of sessions is large, hence an
30   approximation is provided.

One of the traffic descriptor in each class is the
activity level α_i, which is the ratio of mean the rate and
the peak rate of a session. Thus, the number of active

sessions from the $\mathbf{N} = (N_1, \quad N_2, \quad ..., \quad N_K)$ ongoing real-time sessions can be calculated using the multi-dimensional binomial distribution as

$$(9) \quad \Pi(\mathbf{n}) = \prod_i \binom{N_i}{n_i} \cdot (\alpha_i)^{n_i} \cdot (1 - \alpha_i)^{N_i - n_i}$$

5   If the mean input rate in a queue exceeds the mean service rate, then the buffer fills up quickly and new packets will be dropped or delayed, which results in violated QoS requirements. If delay requirements are small compared to the length of ON and OFF periods of a session 10   then the buffer sizes are relatively small too. Therefore, we assume that all packets arriving in an overloaded queue are lost. The *ratio of lost packets* expresses the probability that a packet arrives in an overloaded situation. In case of Fair Queuing scheduler fed by on-off 15   processes the loss ratio measure of class *i* served in queue *k* can be calculated as

$$(10) \quad \frac{\displaystyle\sum_{R_k(\mathbf{n}) > C_k(\mathbf{n})} n_i \cdot \Pi(\mathbf{n})}{\displaystyle\sum_{\forall \mathbf{n}} n_i \cdot \Pi(\mathbf{n})}$$

where $R_i(\mathbf{n})$ is the input rate and $C_i(\mathbf{n})$ is the service rate of queue *k* if the number of active sources $\mathbf{n}$ is given. 20   To provide the approximation, firstly, we use the lower bound on the service rate of queue k as

$$(11) \quad C_k(\mathbf{n}) \geq C_k'(\mathbf{n}) = \kappa_k \left(C - \sum_{j \in RT_k} R_j(\mathbf{n}) \right)$$

where $\kappa_k = \dfrac{w_k}{w_k + \sum_{j \in BE} w_j}$, and *BE* and RT_k is the index set of

BE queues and RT queues excluding *k*, respectively. Secondly, 25   we use the fact that the number of active sessions in a class converges to the normal distribution if the number of sessions grows. Thus, the ratio of lost packets can be approximated as

$$(12) \quad \frac{\int\limits_{-\infty}^{\infty}\int\limits_{-\infty}^{\infty} K \int\limits_{\kappa_k \cdot \left(C-\sum_{j\neq l} x_j\right)}^{\infty} x_k \varphi_1(x_1)\varphi_2(x_2) K \; \varphi_K(x_K)\, dx_k dx_1 K\, dx_K}{\int\limits_{-\infty}^{\infty}\int\limits_{-\infty}^{\infty} K \int\limits_{-\infty}^{\infty} x_k \varphi_1(x_1)\varphi_2(x_2) K \; \varphi_K(x_K)\, dx_k dx_1 K\, dx_K}$$

where $\varphi_k(x)=\varphi(x,r_k,v_k)$ is the density function of the normal distribution with mean $r_k=\sum_{j\in Q_k} N_j\rho_j\alpha_j$ and variance $v_k=\sum_{j\in Q_k} N_j\rho_j^2\alpha_j(1-\alpha_j)$, where Q_k is the index set of classes that

5 are served in queue k, and ρ_j is the mean input rate in class j. Reformulation of the approximation leads to closed form:

$$(13) \quad 1-\Phi\left(C,\tilde{R}_k,\tilde{V}_k\right)+\frac{\tilde{V}_k}{\tilde{R}_k}\cdot\varphi\left(C,\tilde{R}_k,\tilde{V}_k\right)$$

where $\tilde{R}_k=\kappa_k^{-1}r_k+\sum_{j\neq k}r_j$ and $\tilde{V}_k=\kappa_k^{-2}v_k+\sum_{j\neq k}v_j$ are the corrected mean and variance of input rate in queue k.

10

In Figure 8 the simplified flowchart of an exemplary call admission control method is shown. The method consist of an initialization phase 601 and a real-time phase 602. When an admission request 64 is invoked, prior to the

15 initialization phase 601 a decision is made whether the descriptor of a traffic class changed 63. If yes, initialization phase 601 is invoked and in step 61 coefficients of the approximating hyperplanes are computed and in step 62 stored. This phase is repeated when the

20 descriptor of a traffic class changes 63, which usually happens when nodes are configured or reconfigured, otherwise in the real-time phase 602 two decisions are made. A traffic mix is admissible 67, if for each real-time traffic class both the stability 65 and the delay constraints 66 are

25 fulfilled. If not, the admission is rejected 68. In another implementation stability and delay check are performed simultaneously.

Initialization   method   waits   the   following   input
parameters:

- link capacity C
- number of real-time classes K
- number of real-time queues in the FQ scheduler L
- traffic descriptors of the real-time classes $(T_i, b_i, s_i, \alpha_i)$ where

    - T_i is the Transmission Time Interval (TTI)
    - b_i is the packet size
    - s_i the size of the last segment (if no segmentation is applied for that class, then $s_i = b_i$)
    - α_i is the activity level

- QoS requirements of real-time classes (d_i, ε_i), where

    - d_i is the delay requirement assigned to that node
    - ε_i is the tolerance level against QoS violations

- assigned DiffServ queue D_i for each traffic class
- weights w_i assigned to the queues in the FQ scheduler

    As output, the initialization method provides the following arrays

- $H(i)$: the number of approximating hyperplanes for RT class i (1D array containing integer values, $H(i) \leq 2^{L-1}$)

- $E(M,j,i)$: effective bandwidth value for a class j session expressed in number of class i sessions in scheduling model M (3D array containing floating point values; $M \leq H(i)$ and $i, j \leq K$)

- $P(M,i)$: maximal number of class i sessions in scheduling model M if no ongoing sessions from other classes are present (2D array containing floating point values; $M \leq H(i)$ and $i \leq K$)

- $K(q)$: capacity share of real-time queue q if each other real-time queues are empty (1D array containing floating point values, $q \leq L$)

5   An exemplary pseudo code for the implementation of the initialization phase can be as follows:

1. For each real-time class i : H(i) := 0
2. For each real-time class i : call procedure
10   SpModel(i,L, saturated)
3. For each real-time queue q: call procedure CapacityShare(q)
4. EXIT with H, E, P and K

15   Procedure SpModel(i, j, saturated)

1.   If j = 0, then call Calculation(i, saturated). Then EXIT

2.   If queue(j) is non-real-time, then saturated(j) := true and call SpModel(i,j-1, saturated). Then
20   EXIT.

3.   Else if j = D_i, then saturated(j) := false, and call SpModel(i,j-1, saturated). Then EXIT

4.   saturated(j) := TRUE and call SpModel(i,j-1, saturated)

25   5.   saturated(j) := FALSE and call SpModel(i,j-1,saturated)

6.   EXIT

Procedure Calculation(i, saturated)

30   1. - $r := \dfrac{w_i}{w_i + \sum_{j \in saturated} w_j}$

2. c := C * r

```
3. P(H(i), i) := call FIFOmax(c, T_i, b_i, α_i , d
   i- b_i /c, ε_i)
4. If P(H(i),i) = 0, then EXIT.
5. j := 1
6. If j = i then E(H(i),i,i) := 1
7. Else If saturated(D_j) then E(H(i),j,i) := 0
8. Else If D_j = D_i then
   x :=  call FIFOmax(c, T_j, b_j, α_j , d_i- b_i /c,
   ε_i)
   If x = 0 then E(H(i),j,i) := P(H(i),i) + 2
   Else E(H(i),j,i) := P(H(i),i) / x
9. Else
   x := call SPmax(c, T_j, b_j*r, α_j , b_i, s_i, d_i,
   ε_i)
   If x = 0 then E(H(i),j,i) := P(H(i),i)+2
   Else E(H(i),j,i) := P(H(i),i) / x
10.    j := j +1
11.    If  j ≤ L then GOTO 6;
12.    H(i) := H(i) + 1
13.    EXIT
```

Function FIFOmax(c, T, b, α, d, ε)

```
1. N := 1
```

$$2. \quad r := \sum_{n=0}^{N} Q_{T,c}^{n}(d) \cdot \Pi(n), \text{ where}$$

$Q_{T,c}^{n}(x)$ is defined below

$$\Pi(n) = \binom{N}{n} \cdot (\alpha)^{n} \cdot (1-\alpha)^{N-n}$$

```
3. If r > ε then RETURN N-1;
4. N := N + 1
5. GOTO 2;
```

<u>Function SPmax(c, T, b_H, α_H, b_L, s_L, d_L, ε_L)</u>

1. N := 1

2. $r := \sum_{n=0}^{N} D_{T,c}^n(d) \cdot \Pi(n)$, where

   $D_{T,c}^n(x)$ is defined below

   $\Pi(n) = \binom{N}{n} \cdot (\alpha)^n \cdot (1-\alpha)^{N-n}$

3. If $r > \varepsilon_L$ then RETURN N-1;

4. N := N + 1

5. GOTO 2;

<u>Procedure CapacityShare(q)</u>

1. $\kappa_q = \dfrac{w_q}{w_q + \sum_{j \in BE} w_j}$, where

   BE is the set if best effort (non-real-time) queues

2. EXIT

It should be noted that pseudo-code above recalculates each element in the output arrays. However, during normal operation only elements assigned to new classes should be computed. In function FIFOmax, probability of violating delay requirements can be calculated using approximations instead of direct calculation.

The superposition of N independent periodic sources with the same period T in a FIFO buffer, i.e. having a single QoS class, has already been studied in a number of contributions before. The exact closed form solution for complementary distribution function of the virtual waiting time is known was as:

(5) $Q_{T,c}^N(x) = \sum_{i=\lceil \frac{xC}{b} \rceil}^{N} \binom{N}{i} \left(\frac{ib}{CT} - \frac{x}{T}\right)^i \left(1 - \frac{ib}{CT} + \frac{x}{T}\right)^{N-i-1} \left(1 - \frac{Nb}{CT} + \frac{x}{T}\right),$

where N is the number of sources, T is the period length, C is the server capacity, b is the packet size, and $\lceil . \rceil$ denotes the upper integer value.

Expression (5) can be easily extended to multiple
5   traffic classes that are heterogeneous in packet size, however, extension to heterogeneous period length requires approximations.

Despite the simplicity of (5), the maximal number of schedulable users for a given delay-loss constraint cannot
10 be expressed from the formulae in closed form. There are a simple and invertable approximation for the distribution of the virtual waiting time based on the Brownian bridge approximation. The complementary distribution function of the virtual waiting time according to the Brownian bridge
15 approximation for multiple classes is:

$$(6) \quad Q_{T,C}^{N_1,N_2,\ldots,N_k}(x) \approx \exp\left\{-\frac{2xC}{\sum_i N_i b_i^2}\cdot\left((x+T)C-\sum_i N_i b_i\right)\right\},$$

where the new notations are: k – the number of traffic classes, N_i – the number of sources in class i and b_i – the packet size in class i. The distribution of the queuing
20 delay is expressed by (5) and (6), thus the statistical delay requirement for a class j packet is met if the following inequality is fulfilled:

$$(7) \quad Q_{T,C}^{N_1,\ldots,N_j-1,\ldots,N_k}\left(d_j\right)<\varepsilon_j$$

where d_j is the maximum queuing delay of a class j
25 packet and ε_j is the tolerated delay violation probability for class j.

Search method in functions FIFOmax and SPmax can be much more effective than one in the pseudo-code above (e.g.
30 binary search).

The principles derived for FIFO scheduling can be extended to Strict Priority (SP) scheduling. In this model, independent periodic sources are grouped into traffic classes according to their traffic parameters. Each traffic class has its own queue in the investigated scheduling architecture, and a strict priority order is given for the queues.

For the distribution of the queuing delay was shown that in a non-preemptive strict priority system that includes one low priority periodic source with period length T, packet size b_L, N high priority sources with period length T and packet size b_H, the complementary distribution function of the delay between the arrival of the low priority packet and the beginning of its service is

$$(8) \quad Q_{T,C}^N(x) = \sum_{n=\left\lceil \frac{xC}{b_H} \right\rceil}^{N} \binom{N}{n} (\rho_n)^{n-1}(1-\rho_n)^{N-n-1}(1-\rho)\left(\rho_n - \frac{x}{T}\right),$$

where $\rho_n = \dfrac{nb_H}{CT}$ and $\rho = \dfrac{Nb_H}{CT}$.

In Figure 9 an exemplary embodiment of the real-time phase is illustrated. As it is seen stability 902 and delay 901 can be checked parallel when a CAC request 90 is invoked. During stability check 902, stability of each real-time queue is tested, by estimating the loss ratio by the LossRatio(N,q) function and comparing the minimal allowed tolerance level in that queue. In the step 98 it is determined whether each real-time queue (RTQ) is checked 98. If is not, the next RTQ is selected 99. In step 100 it is determined that LossRation in that RTQ is acceptable. If is not, traffic mix is rejected 97.

Delay check 901 is done by using the proposed linear approximation method for each real-time traffic class. A

traffic mix fulfills the delay requirement of a class, if it is below at least one approximating hyperplane in the space of number of sessions. This is tested by comparing the scalar product of the traffic mix vector and the normal vector of the hyperplane to P(s,i). Delay check 901 is successful, if at least one appropriate hyperplane is found for each class. In step 91 it is determined that each traffic class is checked. If is not, next traffic class is selected. In step 93 it is determined that each hyperplane of that traffic class is checked. If is not, then it is determined 95 whether N is below of that hyperplane.

Traffic mix is accepted 96, if both stability check for each real-time queue and delay check for each real-time class are successful otherwise traffic mix is rejected 97.

As an exemplary implementation can be as follows:

<u>Scalar product E*N^T (i,s,N)</u>

1.   x := 0; j := 1

2.   x := x + N(j)*E(s,j.i)

3.   j := j+1

4.   If j ≤ K then GOTO 2

5.   Return x;

<u>Function LossRatio(N,q)</u>

1. R := 0, V:= 0

2. For each real-time class i

$$r := N_i \alpha_i \frac{b_i}{T_i} \; ; \quad v := N_i \alpha_i (1-\alpha_i) \frac{b_i^2}{T_i^2}$$

If D$_i$ = q then

R = R + r/K(q); V = V + v/(K(q))^2;

Else

```
          R = R + r; V = V + v;
    3. If V = 0 Then
       If R > C Then Return 1
       Else Return 0
5   4. Else
       Return 1-Φ(C,R,V) + V/R * φ(C,R,V)
```

The symbols $\varphi(x,\mu,\sigma^2)$ and $\Phi(x,\mu,\sigma^2)$ refers to the density and cumulative distribution functions of the normal distribution with mean μ and variance σ^2.

10   Efficiency of the CAC method can be improved by adjusting the sequence of checking stability of queues, and sequence of checking the delay requirements of classes, as well as the sequence of hyperplanes to which the traffic mix is compared. Adjustment can be done by configuration (e.g.

15   based on call level parameters of the load) or by applying an LRU (Last Recently Used) sorting.

        Although several embodiments of the present invention have been illustrated in the accompanying Drawings and

20   described in the foregoing Detailed Description, it is understood that the invention is not limited to the embodiments disclosed, but is capable of numerous rearrangements, modifications, and substitutions without departing from the scope of the invention as set forth and

25   defined by the following claims.